CLAIMS

1. A system for debugging a computer application that employs rights-managed (RM) content, the application normally being instantiated in an isolated process with a trusted component for performing RM services for the application including ensuring that a debugger is not monitoring the isolated process, the system comprising:

a first, non-isolated process having the application and a shell version of the trusted component, such shell version of the trusted component receiving each request by the application for RM services, the first process for being monitored by the debugger and the shell version of the trusted component in the first process being unconcerned whether the debugger is monitoring the first process; and

a second, isolated process separate from the first process and having a debugging version of the trusted component, the shell version of the trusted component in the first process forwarding the received request to the debugging version of the trusted component in the second process, such debugging version of the trusted component in the second process acting upon the request from the application in the first process, the debugging version of the trusted component in the second process also ensuring that the debugger is not monitoring the second process, the debugging version of the trusted component in the second process being unconcerned whether the debugger is monitoring the first process,

whereby the debugger may monitor the application and the first process even as the application and first process are employing the RM content.

2. The system of claim 1 wherein the debugging version of the trusted component in the second process based on the acted-upon request

returns appropriate data, if any, back to the application in the first process by way of the shell version of the trusted component in such first process.

3. The system of claim 1 further comprising the debugger monitoring the first process and the application thereof.

4. The system of claim 1 wherein the application in the first process includes an executable and the debugging version of the trusted component in the second process includes a library of services available to the executable corresponding to the application.

5. The system of claim 1 wherein the debugging version of the trusted component in the second process is a library and the second process requires an executable, and wherein the second process further has a shell executable.

6. The system of claim 5 wherein the shell executable includes no substantial functionality.

7. The system of claim 1 wherein the shell version of the trusted component in the first process includes no substantial functionality.

8. The system of claim 1 further comprising a debugging manifest corresponding to the debugging version of the trusted component in the second process, the debugging manifest including information therein relevant to the debugging version of the trusted component and the second process, the debugging version of the trusted component during operation thereof referring to the debugging manifest for the information therein.

9. The system of claim 8 wherein the information in the debugging manifest is selected from a group consisting of a description of an environment of the second process, a digital signature with a verifying certificate

chain, keys that are to be employed to verify constituent elements of the second process, procedures to be followed by the debugging version of the trusted component, and combinations thereof.

10. The system of claim 8 wherein the information in the debugging manifest restricts the debugging version of the trusted component in the second process to allowing the application to employ debug RM content only and not normal RM content.

11. The system of claim 10 wherein debug RM content has a digital signature associated therewith from a debug RM server and normal RM content does not have such a digital signature associated therewith from a debug RM server.

12. The system of claim 10 wherein debug RM content has a digital signature associated therewith that derives from a debug root key and normal RM content does not have such a digital signature associated therewith that derives from a debug root key.

13. A method for debugging a computer application that employs rights-managed (RM) content, the application normally being instantiated in an isolated process with a trusted component for performing RM services for the application including ensuring that a debugger is not monitoring the isolated process, the method comprising:

the application and a shell version of the trusted component being instantiated in a first, non-isolated process, the first process for being monitored by the debugger and the shell version of the trusted component in the first process being unconcerned whether the debugger is monitoring the first process;

a debugging version of the trusted component being instantiated in a second, isolated process, the debugging version of the trusted component in the second process ensuring that the debugger is not monitoring the

second process, the debugging version of the trusted component in the second process being unconcerned whether the debugger is monitoring the first process,

the application in the first process receiving a request to render RM content and in turn requesting the shell version of the trusted component to assist in decrypting and rendering the content;

the shell version of the trusted component in the first process forwarding the request to the debugging version of the trusted component in the second process;

the debugging version of the trusted component in the second process determining that the RM content is allowed to be rendered; and

the debugging version of the trusted component in the second process decrypting the RM content and returning same to the application in the first process for rendering thereby,

whereby the debugger may monitor the application and the first process even as the application and first process are employing the RM content.

14. The method of claim 13 wherein the debugging version of the trusted component in the second process determining that the RM content is allowed to be rendered comprises the debugging version of the trusted component determining that the RM content is allowed to be rendered based on a license corresponding thereto.

15. The method of claim 13 wherein the debugging version of the trusted component and a debugging manifest corresponding to the debugging version of the trusted component are instantiated in the second, isolated process, the debugging manifest including information therein relevant to the debugging version of the trusted component and the second process, and wherein the debugging version of the trusted component in the second process determining that the RM content is allowed to be rendered comprises the debugging version of the trusted component determining that the RM content is allowed to be rendered based on the information in the debugging manifest.

16. The method of claim 15 wherein the information in the debugging manifest restricts the debugging version of the trusted component in the second process to allowing the application to employ debug RM content only and not normal RM content.

17. The method of claim 16 wherein debug RM content has a digital signature associated therewith from a debug RM server and normal RM content does not have such a digital signature associated therewith from a debug RM server.

18. The method of claim 16 wherein debug RM content has a digital signature associated therewith that derives from a debug root key and normal RM content does not have such a digital signature associated therewith that derives from a debug root key.

19. The method of claim 13 wherein the debugging version of the trusted component in the second process returns the decrypted RM content to the application in the first process by way of the shell version of the trusted component in such first process.

20. The method of claim 13 further comprising the debugger monitoring the first process and the application thereof.

21. The method of claim 13 wherein the debugging version of the trusted component in the second process is a library and the second process requires an executable, and wherein the debugging version of the trusted component and a shell executable are instantiated in the second, isolated process.

22. The method of claim 21 comprising instantiating the shell executable with no substantial functionality.

23. The method of claim 13 comprising instantiating the shell version of the trusted component in the first process with no substantial functionality.